Reaction Systems, Transition Systems, and Equivalences

Jetty Kleijn[1], Maciej Koutny[2], Łukasz Mikulski[3], and Grzegorz Rozenberg[1,4]

[1] LIACS, Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands
{h.c.m.kleijn,g.rozenberg}@liacs.leidenuniv.nl
[2] School of Computing, Newcastle University, NE1 7RU, UK
maciej.koutny@ncl.ac.uk
[3] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, 87-100 Toruń, Poland
lukasz.mikulski@mat.umk.pl
[4] Department of Computer Science, University of Colorado at Boulder
430 UCB Boulder, CO 80309-0430, U.S.A.

Abstract. Reaction systems originated as a formal model for processes inspired by the functioning of the living cell. The underlying idea of this model is that the functioning of the living cell is determined by the interactions of biochemical reactions and these interactions are based on the mechanisms of facilitation and inhibition. Since their inception, reaction systems became a well-investigated novel model of computation. Following this line of research, in this paper we discuss a systematic framework for investigating a whole range of equivalence notions for reaction systems. Some of the equivalences are defined directly on reaction systems while some are defined through transition systems associated with reaction systems. In this way we establish a new bridge between reaction systems and transition systems. In order to define equivalences which capture various ways of interacting with an environment, we also introduce models of the environment which evolve in a finite-state fashion.
Keywords: reaction system, living cell, natural computing, interactive computation, transition system, behavioural equivalences, bisimulation, context controller

1 Introduction

Natural computing is an interdisciplinary research area concerned with human-designed computing inspired by nature and with computing taking place in nature (see, e.g., [22] and [29]). The former strand investigates models and computational techniques inspired by nature, while the latter investigates, in terms of information processing, phenomena taking place in nature. Clearly, the two research strands are not disjoint.

Biomolecular computation is a topic of intense research in natural computing. Within the former strand this research focuses on constructing, either in vitro or in vivo, various building blocks of computing devices, as well as on designing novel, nature inspired algorithms. Within the latter strand this research is concerned with establishing how biocomputations drive natural processes.

The original motivation behind reaction systems (see, e.g., [17, 8, 12]), the subject of this paper, falls into this second strand of research. Reaction systems were proposed as a formal model for the investigation of the functioning of the living cell, where this functioning is viewed in terms of formal processes resulting from interactions between biochemical reactions taking place in the living cell. The key feature of these interactions is that they are driven by two mechanisms, facilitation and inhibition: the (products of the) reactions may facilitate or inhibit each other. The basic model of reaction systems abstracts from various (technical) features of biochemical reactions to such extent that it becomes a qualitative rather than a quantitative model. However, it takes into account the basic bioenergetics (flow of energy) of the living cell, see [24], and it also takes into account that the living cell is an open system and its behaviour is influenced by its environment.

Although the model of reaction systems was inspired by biology, the research on reaction systems is driven by both biological considerations (see, e.g., [6, 18, 19, 2, 3, 7, 10]) and considerations concerned with the understanding of computations taking place in reaction systems which formalise dynamic processes instigated by the interactions of biochemical reactions in the living cell (see, e.g., [16, 13, 14, 20, 8, 11, 30–32]). As a matter of fact, reaction systems turned out to be an interesting novel model of interactive computation.

This paper follows this computational line of research and its goal is to present a systematic framework for considering various notions of behavioural equivalence for reaction systems. Within this framework we consider equivalences defined directly on reaction systems as well as equivalences defined through transition systems associated with them. In this way we establish a new bridge to transition systems which are central constructs of theoretical computer science used extensively to investigate behaviours of dynamic processes (see, e.g., [1, 4, 23, 26]).

By associating various sorts of transition systems with reaction systems and then by considering various notions of equivalence for these transition systems, we capture a wide range of properties of dynamic processes taking place in reaction systems.

Reaction systems are models of interactive computations, the interaction of a reaction system with its environment constitutes a key feature of this model. In order to account for these interactions in our framework, we introduce models of environment that evolve in a finite-state fashion.

The paper is self-contained in the sense that (1) it introduces basic notions concerning reaction systems and their dynamic processes (together with the original biological motivation), and (2) it introduces basic notions concerning transition systems.

The paper is organised as follows.

In Section 2 and Section 3 we recall basic notions of reaction systems to be used in this paper, also recalling the original biological motivation.

In Section 4 we consider two basic equivalences defined directly on reaction systems:

- the standard functional equivalence which is local in the sense that it compares the effect of two systems on individual states (there are finitely many of them), and
- the new process equivalence which is global in the sense that it compares the sets of processes of two systems (there are infinitely many of them).

Both equivalences considered in Section 4 are quite restrictive, and so in Section 5 we consider two ways of making these equivalences less restrictive:

- by comparing two systems only w.r.t. processes that begin in a designated set of possible initial states, and
- by making only some entities visible to an outside observer.

In Section 6 we recall basic notions of transition systems to be used in this paper, and then in Section 7 we establish a method of associating transition systems with reaction systems.

Reaction systems are models of interactive computing, the interaction of a system with its environment is the central feature of processes of reaction systems. As a preparatory step for incorporating this interaction in equivalence notions, in Section 8 we formalise an intuitive concept of a 'finite-state' environment, i.e., an environment which evolves in a finite-state fashion. It is an intermediate between the two sorts of environments mostly considered in the literature: all possible contexts are allowed and no context is allowed (i.e., a reaction system is considered to be a 'closed' system). The resulting formal constructs are called *context controllers* and they can control the environment either totally independently of reaction systems (operating in it) or in a fashion that is dependent on (aware of) the current states of reaction systems. Incorporating context controllers into the behaviour of reaction systems leads to new definitions of their processes.

In Section 9 we consider equivalences defined not directly on reaction systems, but rather through transition systems associated with them.

The notion of bisimulation is a central notion of the theory of concurrent systems [26, 27, 35] for comparing their dynamic behaviours. In Section 10 we transfer this notion to reaction systems using their representations by transition systems.

The discussion in Section 11 concludes this paper.

2 Reactions and Reaction Systems

In this section and in Section 3, we recall the basic notions of reaction systems needed in this paper. Most of them are taken from [5, 12, 16].

The formal notion of a reaction reflects the basic intuition of a biochemical reaction – it will take place if all its reactants are present and none of its inhibitors is present; when a reaction takes place it creates its products.

Definition 1. *A* reaction *is a triple* $b = (R, I, P)$ *such that* R, I, P *are finite nonempty sets with* $R \cap I = \varnothing$.

The sets R, I, P are called the *reactant set of* b, the *inhibitor set of* b, and the *product set of* b, respectively – they are also denoted as R_b, I_b, and P_b, respectively. If $R, I, P \subseteq Z$ for a finite set Z, then we say that b is a *reaction in* Z and we use $rac(Z)$ to denote the set of all reactions in Z – note that $rac(Z)$ is finite.

Since R and I are nonempty and disjoint, a finite set Z as above must have at least 2 elements – we refer to such finite sets as *background sets*.

To describe the effect of a set of reactions on a state (of a biochemical system), we first define the effect of a single reaction.

Definition 2. *Let Z be a background set, let $X \subseteq Z$, and let $b \in rac(Z)$. Then b is* enabled by X, *denoted by $en_b(X)$, if $R_b \subseteq X$ and $I_b \cap X = \varnothing$. The* result of b on X, *denoted by $res_b(X)$, is defined by $res_b(X) = P_b$ if $en_b(X)$, and $res_b(X) = \varnothing$ otherwise.*

Here the finite set X is a formal representation of a state (i.e., the set of biochemical entities currently present in the given biochemical system). Then b is enabled by X if X separates R_b from I_b, meaning that all reactants from R_b are present in X and none of the inhibitors from I_b is present in X. When b is enabled by X, it contributes its product P_b to the successor state; otherwise it does not contribute anything to the successor state.

The effect of a set of reactions on a state is cumulative, which is formally defined as follows.

Definition 3. *Let Z be a background set, let $X \subseteq Z$, and let $B \subseteq rac(Z)$. The* result of B on X, *denoted by $res_B(X)$, is defined by $res_B(X) = \bigcup\{res_b(X) \mid b \in B\}$.*

Note that if the transition from a current state X to its successor is determined only by the results of reactions (i.e., there is no influence of the environment, which we will consider later on), then the successor state consists only of the entities produced by the reactions enabled in the current state. This implies that in the transition from the current state to its successor state an *entity from X vanishes unless it is sustained/produced by a reaction.* This is the *non-permanency property* which reflects the basic bioenergetics of the living cell: *without a flow/supply of energy the living cell disintegrates, but the use/absorption of energy by the living cell is realised through biochemical reactions* (see, e.g., [24]). Thus an entity vanishes within *one* state transition unless it is produced by a reaction. This non-permanency property is a major difference between reaction systems and models considered in the theory of computation (see, e.g., [1] and [21]). Also finite duration of entities in reaction systems – corresponding to their presence in several consecutive states – which takes into account decay time, is considered in the literature (see, e.g., [6]).

There is another notable aspect of Definition 3. If a, b are two reactions from B enabled by X, then both of them will take place even if $R_a \cap R_b \neq \varnothing$. Hence there is no notion of conflict between reactions even if they need to share reactants. This is the property of the *threshold nature of resources: either an entity*

is available and then there is enough of it, or it is not available. The threshold nature of resources (no conflict) property is a major difference with structural models of concurrency, such as, e.g., Petri nets [28]. This property reflects the *level of abstraction* adopted for the formulation of the basic model of reaction systems: one does not count concentrations of entities/molecules to infer from these which reactions can/will be applied. One operates on a higher level of abstraction: assuming that the cell is running/functioning, the aim is to understand the ongoing processes. This level of abstraction can be compared with the level of abstraction of the standard models of computation in computer science, such as Turing machines and finite automata. These standard models turned out to be very successful in understanding computational processes running on electronic computers, and yet nothing in these models takes into account the electronic/quantitative properties of the underlying hardware. It is simply assumed that the underlying electronics/hardware functions 'well' and then the goal is to understand processes running on (implemented by) this hardware. Similarly, we want to understand the processes resulting from interactions in the living cell and at this stage we abstract from the underlying 'hardware properties' of the living cell. Thus: *the basic model of reaction systems is qualitative rather than quantitative – in particular, there is no counting.*

With the formal notion of a reaction and its effect on states established, we can now proceed to formally define reaction systems as an abstract model of the interactions of biochemical reactions in the living cell.

Definition 4. *A* reaction system *is an ordered pair* $\mathcal{A} = (S, A)$*, where* S *is a background set and* A *is a nonempty finite subset of* $rac(S)$*.*

The set S is called the *background set of* $\mathcal{A}$, and its elements are called the *entities of* $\mathcal{A}$ – they represent molecular entities (e.g., atoms, ions, molecules) that may be present in the states of the biochemical system (e.g., the living cell). The set A is called the *set of reactions of* $\mathcal{A}$; clearly A is finite (as S is finite).

The subsets of S are called the *states of* $\mathcal{A}$. Given a state $X \subseteq S$, the *result of* $\mathcal{A}$ on X, denoted by $res_{\mathcal{A}}(X)$, is defined by $res_{\mathcal{A}}(X) = res_A(X)$.

Thus a reaction system is essentially a set of reactions. We also specify the background set which consists of entities needed for defining the reactions and for reasoning about the reaction system (see the definition of an interactive process below). There are no 'structures' involved in reaction systems (such as, e.g., the tape of a Turing machine). Finally, note that this is a *strictly finite model* – the size of any state is a priori restricted (by the size of the background set).

3 Processes of Reaction Systems

The model of reaction systems formalises the 'static structure' of the living cell as the set of all reactions that can take place in the cell (together with the set of underlying entities). Since the original motivation behind this model was to understand the functioning of living cells, one is really interested in the processes instigated by the interactions of these reactions. They are formalised as follows.

Definition 5. *Let $\mathcal{A} = (S, A)$ be a reaction system.*

An interactive process *in $\mathcal{A}$ is an ordered pair $\pi = (\gamma, \delta)$ of finite sequences such that $\gamma = C_0, \ldots, C_n$ and $\delta = D_0, \ldots, D_n$, for some $n \geq 1$, where $C_0, \ldots, C_n \subseteq S$, $D_0, \ldots, D_n \subseteq S$, and $D_i = res_{\mathcal{A}}(D_{i-1} \cup C_{i-1})$, for all $i \in \{1, \ldots, n\}$.*

The sequence γ is the *context sequence of* π, the sequence δ is the *result sequence of* π, and the sequence $\tau = W_0, \ldots, W_n$, where, for all $i \in \{0, \ldots, n\}$, $W_i = C_i \cup D_i$, is the *state sequence of* π, with $W_0 = C_0 \cup D_0$ the *initial state* of π. We let $STS(\pi)$ denote the state sequence of π. By $PROC(\mathcal{A})$ we denote the set of interactive processes of $\mathcal{A}$ and, for $X \subseteq S$, $PROC_X(\mathcal{A})$ is the set of interactive processes of $\mathcal{A}$ with initial state X.

The dynamic process formalised by an interactive process π begins in its initial state. The reactions of $\mathcal{A}$ enabled by this initial state W_0 produce the result set D_1, which together with the context set C_1 forms the successor state $W_1 = res_{\mathcal{A}}(W_0) \cup C_1$. This formation of successor states is iterated: $W_i = res_{\mathcal{A}}(W_{i-1}) \cup C_i$, resulting in the state sequence $STS(\pi) = W_0, \ldots, W_n$.

An interactive process may be visualised by a three-row representation, where the first row represents the context sets and is labelled by 'γ', the second row represents result sets and is labelled by 'δ', and the third row represents states and is labelled by 'τ'. Such a representation looks as follows:

$$\begin{array}{lccccc} \gamma: & C_0 & C_1 & \ldots & C_{n-1} & C_n \\ \delta: & D_0 & D_1 & \ldots & D_{n-1} & D_n \\ \tau: & W_0 & W_1 & \ldots & W_{n-1} & W_n \end{array} \tag{1}$$

Note that the context sequence γ together with the initial state W_0 uniquely determine π, because γ and D_0 uniquely determine π (through the result function $res_{\mathcal{A}}$). The context sequence formalises the fact that the *living cell is an open system* in the sense that its behaviour is influenced by its environment (the 'rest' of a bigger system).

If, for all $i \in \{0, \ldots, n\}$, $C_i \subseteq D_i$, then we say that π is *context-independent*: whatever C_i adds to D_i, has already been produced by the system (is included in the result D_i) or perhaps C_i adds nothing at all ($C_i = \varnothing$). If π is context-independent, then (in its analysis) we may as well assume that each C_i, for $i \in \{0, \ldots, n\}$, adds nothing, hence the context sequence consists of empty sets C_i only. Clearly, if π is context-independent, then the initial state $W_0 = D_0$ determines π by the repeated application of $res_{\mathcal{A}}$.

When reaction systems were introduced, the definition of an interactive process assumed that $D_0 = \varnothing$. This is not the case anymore and also in this paper this condition is dropped, in this way a suffix (of length at least 2) of an interactive process is also an interactive process. As a consequence we assume that in a context-independent process also $C_0 = \varnothing$.

We will use $CIPROC(\mathcal{A})$ to denote the set of context-independent interactive processes of $\mathcal{A}$ and $CIPROC_{W_0}(\mathcal{A})$ to denote the set of context-independent interactive processes of $\mathcal{A}$ with initial state W_0.

4 Direct Equivalences

In this section we consider equivalences for reaction systems defined directly through their reactions and interactive processes. First we recall the standard functional equivalence that was introduced right from the start in [17] for reaction systems.

Definition 6. *Reaction systems $\mathcal{A}$ and $\mathcal{B}$ with the same background set S are* (functionally) equivalent*, denoted by $\mathcal{A} \sim \mathcal{B}$, if $res_{\mathcal{A}}(X) = res_{\mathcal{B}}(X)$ for all $X \subseteq S$.*

Note that, strictly speaking, we should have written $\sim_S$ in the above definition. However to simplify our notation we will use $\sim$ whenever this will not lead to confusion.

The concept of interactive processes induces in a natural way the following notion of equivalence for reaction systems.

Definition 7. *Reaction systems $\mathcal{A}$ and $\mathcal{B}$ with the same background set are* process equivalent*, denoted by $\mathcal{A} \sim_{PROC} \mathcal{B}$, if $PROC(\mathcal{A}) = PROC(\mathcal{B})$.*

The relation $\sim_{PROC}$ is clearly an equivalence relation. Moreover, it turns out that $\sim$ which is defined locally (just on all subsets of the background set) and $\sim_{PROC}$ which is defined globally (over all, infinitely many, processes) coincide.

Proposition 1. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Then $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if $\mathcal{A} \sim \mathcal{B}$.*

Proof. Suppose that $\mathcal{A} \sim_{PROC} \mathcal{B}$. Thus $PROC(\mathcal{A}) = PROC(\mathcal{B})$. Let $X \subseteq S$ and consider a process $\pi \in PROC(\mathcal{A})$ with initial state X and context sequence $\varnothing, \varnothing$. Thus the result sequence of π is $X, res_{\mathcal{A}}(X)$. Since π is also an interactive process in $\mathcal{B}$, it follows that $res_{\mathcal{B}}(X) = res_{\mathcal{A}}(X)$. Hence, for each $X \subseteq S$, $res_{\mathcal{B}}(X) = res_{\mathcal{A}}(X)$ and consequently $\mathcal{A} \sim \mathcal{B}$.
Now suppose that $\mathcal{A} \sim \mathcal{B}$. Hence, $res_{\mathcal{A}}(X) = res_{\mathcal{B}}(X)$ for all $X \subseteq S$, which directly implies that $\mathcal{A} \sim_{PROC} \mathcal{B}$. □

Similarly, we can also show that one does not need the full information from the interactive processes of reaction systems to establish their (process) equivalence – it suffices to consider only their state sequences.

Proposition 2. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Then $\mathcal{A} \sim \mathcal{B}$ if and only if $STS(PROC(\mathcal{A})) = STS(PROC(\mathcal{B}))$.*

Proof. The only-if-direction follows immediately from Proposition 1.
So let us assume that $STS(PROC(\mathcal{A})) = STS(PROC(\mathcal{B}))$. Let $X \subseteq S$ and consider $\pi \in PROC(\mathcal{A})$ with some initial state W and context sequence $\varnothing, S, X, \varnothing$. Then $STS(\pi) = W, S, X, res_{\mathcal{A}}(X)$. Hence there exists an interactive process $\pi' \in PROC(\mathcal{B})$ such that $STS(\pi') = W, S, X, res_{\mathcal{A}}(X)$. This implies that $res_{\mathcal{B}}(X) \subseteq res_{\mathcal{A}}(X)$. There also exists an interactive process in $\mathcal{B}$ with state sequence

$W, S, X, res_{\mathcal{B}}(X)$. Consequently $W, S, X, res_{\mathcal{B}}(X)$ is the state sequence of an interactive process in $\mathcal{A}$, which implies that $res_{\mathcal{A}}(X) \subseteq res_{\mathcal{B}}(X)$. It follows that $res_{\mathcal{B}}(X) = res_{\mathcal{A}}(X)$. Hence, for each $X \subseteq S$, $res_{\mathcal{B}}(X) = res_{\mathcal{A}}(X)$ and consequently $\mathcal{A} \sim \mathcal{B}$. □

Corollary 1. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set. Then $\mathcal{A} \sim \mathcal{B}$ if and only if $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if $STS(PROC(\mathcal{A})) = STS(PROC(\mathcal{B}))$.*

5 Relaxing direct equivalences

The conditions for equivalence as captured by $\sim$ and $\sim_{PROC}$ may be rightly regarded as too restrictive, because this equivalence entails 'always the same effect' of the reactions of the two reaction systems under consideration. In this section we formulate several different ways of relaxing this constraint.

Initial states

The first idea to relax $\sim_{PROC}$ is to compare reaction systems only with respect to interactive processes that begin in certain designated states.

Definition 8. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Let Z be a nonempty subset of 2^S. Then $\mathcal{A}$ and $\mathcal{B}$ are* process equivalent w.r.t. *Z, denoted by $\mathcal{A} \sim_Z \mathcal{B}$, if $PROC_X(\mathcal{A}) = PROC_X(\mathcal{B})$ for every $X \in Z$.*

The relation $\sim_Z$ is clearly an equivalence relation. Moreover, $\sim_{2^S}$ is nothing but $\sim_{PROC}$. What is perhaps rather unexpected, is that process equivalence is guaranteed for two reaction systems whenever they are equivalent with respect to some nonempty set of initial states. Any subset $Z \subseteq 2^S$ such that $\mathcal{A} \sim_Z \mathcal{B}$ is thus a 'test subset' for process equivalence.

Proposition 3. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Then $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if there exists a nonempty $Z \subseteq 2^S$ such that $\mathcal{A} \sim_Z \mathcal{B}$.*

Proof. Since $\sim_{PROC} = \sim_{2^S}$, we only need to prove the if-direction of the statement. Assume that Z is a nonempty subset of 2^S such that $\mathcal{A} \sim_Z \mathcal{B}$. Let $X \in Z$ and let Y be an arbitrary subset of S. Consider the interactive process π in $\mathcal{A}$ with context sequence $\varnothing, S, Y, \varnothing$ and result sequence $X, res_{\mathcal{A}}(X), \varnothing, res_{\mathcal{A}}(Y)$. From $\mathcal{A} \sim_Z \mathcal{B}$ and $X \in Z$, it follows that π is also an interactive process in $\mathcal{B}$ and so $res_{\mathcal{A}}(Y) = res_{\mathcal{B}}(Y)$. Hence, $res_{\mathcal{B}}(Y) = res_{\mathcal{A}}(Y)$, for each $Y \subseteq S$, and consequently $\mathcal{A} \sim \mathcal{B}$. Thus, by Proposition 1, $\mathcal{A} \sim_{PROC} \mathcal{B}$. □

Thus we may conclude that restricting the set of potential initial states has no effect on the equivalences considered so far.

Corollary 2. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Then $\mathcal{A} \sim \mathcal{B}$ if and only if $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if $\mathcal{A} \sim_Z \mathcal{B}$ for some nonempty $Z \subseteq 2^S$.*

At this point it is worthwhile to observe that also in the case of a restricted set of potential initial states, the state sequences of those interactive processes that are taken into consideration are sufficient to establish (process) equivalence. In fact, even one initial state is sufficient.

Proposition 4. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Let $X \subseteq S$ be such that $STS(PROC_X(\mathcal{A})) = STS(PROC_X(\mathcal{B}))$. Then $\mathcal{A} \sim \mathcal{B}$.*

Proof. The statement can be proved along the lines of the proof of the if-direction of Proposition 2 where the choice of an initial state was an arbitrary one. □

In combination with the definition of state sequences of processes, Proposition 3 and Corollary 2, this leads to the following result.

Corollary 3. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Then $\mathcal{A} \sim \mathcal{B}$ if and only if $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if $STS(PROC_X(\mathcal{A})) = STS(PROC_X(\mathcal{B}))$ for some $X \subseteq S$.*

Observable entities

Another possibility to relax $\sim_{PROC}$ is to relate it to a subset of the background set of the reaction systems under consideration. Intuitively, such subset is the part of the background set that is 'visible' to an observer.

First we need some notation.

As usual, for any set U, we use U^* to denote the set of finite sequences of elements from U; the empty sequence is denoted by λ.
Let U and W be finite sets such that $U \subseteq W$. The projection function $proj_{W,U}$ from $(2^W)^*$ onto $(2^U)^*$ is defined as follows.

- For $W' \subseteq W$, $proj_{W,U}(W') = W' \cap U$.
- This is lifted to sequences of sets in a homomorphic way: $proj_{W,U}(W_1W_2 \ldots W_m) = proj_{W,U}(W_1)proj_{W,U}(W_2) \ldots proj_{W,U}(W_m)$, for every sequence $W_1 \ldots W_m$ with $m \geq 1$ and $W_i \subseteq W$, for $i \in \{1, \ldots, m\}$, and $proj_{W,U}(\lambda) = \lambda$.

In what follows, we may omit the subscript W whenever it is clear from the context – then we simply write $proj_U$.

Definition 9. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S and let $Y \subseteq S$. Then $\mathcal{A}$ and $\mathcal{B}$ are Y-*projection equivalent*, denoted by $\mathcal{A} \sim^Y B$, if $proj_Y(STS(PROC(\mathcal{A}))) = proj_Y(STS(PROC(\mathcal{B})))$.*

The relation $\sim^Y$ is clearly an equivalence relation. Note that all reaction systems over the background set S are $\varnothing$-projection equivalent. Moreover, by Corollary 1, $\sim^S$ is nothing but $\sim_{PROC}$.

Proposition 5. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S, and let $X \subseteq Y \subseteq S$. Then $\mathcal{A} \sim^Y \mathcal{B}$ implies $\mathcal{A} \sim^X \mathcal{B}$.*

Proof. Follows directly from the definitions. □

It should be noted here, that in contrast to process equivalence (with respect to some set of designated initial states), projection equivalence is phrased in terms of state sequences rather than the interactive processes themselves. A main reason for doing this is that projection of interactive processes would also involve a projection of the context (thrown in by the environment).

Initial states and observable entities

To conclude this section, we define an equivalence relation for reaction systems by combining the restriction to designated initial states and the projection onto a subset of the background set.

Definition 10. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Let Z be a nonempty subset of 2^S and let $Y \subseteq S$. Then $\mathcal{A}$ and $\mathcal{B}$ are Y-*projection equivalent w.r.t. *Z, denoted $\mathcal{A} \sim_Z^Y \mathcal{B}$, if $proj_Y(STS(PROC_X(\mathcal{A}))) = proj_Y(STS(PROC_X(\mathcal{B})))$, for every $X \in Z$.*

The relation $\sim_Z^Y$ is an equivalence relation and $\sim_{2^S}^S = \sim_{PROC}$.

Proposition 6. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S. Let $X \subseteq Y \subseteq S$ and let $\varnothing \neq W \subseteq Z \subseteq 2^S$. Then $\mathcal{A} \sim_Z^Y \mathcal{B}$ implies $\mathcal{A} \sim_W^X \mathcal{B}$.*

Proof. Follows directly from the definitions. □

6 Transition systems

Transition systems are a central and important model in the theory of computation. They form an established way ([1, 4, 23]) of capturing the behaviour of a dynamic system. Often infinite behaviour (e.g., given in the form of a language consisting of sequences of action symbols) can be represented by a finite transition system. Transition systems have been used as a semantical model for both sequential and concurrent systems.

Transition systems provide a means of defining and checking behavioural equivalence of different systems. There are several different ways in which behavioural equivalence of two transition systems can be expressed, depending on the application area and the desired degree of behavioural closeness. One possibility is to consider two transition systems equivalent if the languages they generate are the same ([1, 23]). Another, much more demanding, possibility is to consider two transition systems equivalent if there is a bisimulation relation between their states which is preserved by simulating transitions ([26, 27, 35]).

After introducing transition systems (in this section) and relating reaction systems and transition systems (in the next section), we will consider bisimulation-based notions of equivalence for reaction systems.

It should be mentioned here that there is a *direct* relationship between reaction systems and transition systems that has been studied in the literature, see e.g., [16, 20]. One can simulate finite transition systems by reaction systems and one can simulate reaction systems by finite transition systems, see e.g., [5, 16]. Also the state space of a rs $\mathcal{A} = (S, A)$ is a finite directed graph (transition system) with states of $\mathcal{A}$ and its nodes and edges determined either by the result function $res_{\mathcal{A}}$ (in case of context-independent interactive processes) or by the joint effect of $res_{\mathcal{A}}$ and the context sets (in case of unrestricted interactive processes). The isomorphism problem for these graphs and its relationship to the functional equivalence was studied in [20].

In a nutshell, a transition system is a directed graph, where the nodes represent global (system) states and the arcs (transitions) can be labelled with information about the actions/conditions associated with the changes of the state of the system. An evolution of the system (from a certain state) corresponds to a directed path through the graph from the node corresponding to the given state. Thus evolutions can be understood and analysed by looking at ordered sequences of states and (labeles of) transitions. Transition systems may be 'initialised', which means that they have a distinguished initial state from which all possible evolutions of the system start.

Definition 11.

1. *A* transition system *is an ordered pair* $\mathcal{T} = (Q, V)$*, where* Q *is a finite set and* $V \subseteq Q \times Q$*.*
2. *An* initialised transition system *is a triple* $\mathcal{T} = (Q, V, q_0)$ *such that* (Q, V) *is a transition system and* $q_0 \in Q$*.*

Definition 12. *Let* L *be a finite set.*

1. *A* labelled transition system *over* L *is an ordered pair* $\mathcal{T} = (Q, V)$*, where* Q *is a finite set and* $V \subseteq Q \times L \times Q$*.*
2. *An* initialised labelled transition system *over* L *is a triple* $\mathcal{T} = (Q, V, q_0)$ *such that* (Q, V) *is a labelled transition system over* L *and* $q_0 \in Q$*.*

The set of state sequences of a transition system $\mathcal{T} = (Q, V)$, denoted by $sts(\mathcal{T})$, is the set of all finite sequences $\theta = q_0, q_1, \ldots, q_n$ $(n \geq 0)$ of elements of Q such that $(q_i, q_{i+1}) \in V$ for all $0 \leq i < n$. Moreover, we denote by $sts_q(\mathcal{T})$ the set of all state sequences of $\mathcal{T}$ which start from a given state q.

7 Associating transition systems with reaction systems

In order to capture a range of behavioural equivalences for reaction systems, we first introduce a representation of their interactive processes as well as their context-independent interactive processes in terms of a suitable class of transition systems. Then, we will also associate other transition systems with a given reaction system, reflecting different views one might adopt when observing its behaviour in context.

The first part of the next definition assumes – similarly to the assumption underlying the definition of interactive processes – that the environment can provide any subset of the background set at any time of the execution of a reaction system. The transitions of the transition system associated with a reaction system capture the change of a state of the reaction system resulting from the reactions taking place at that state and the context thrown in by the environment. The second part of the definition assumes that the environment never provides any entities.

Definition 13. *Let $\mathcal{A} = (S, A)$ be a reaction system.*

1. *The* (full) transition system *of $\mathcal{A}$ is an ordered pair $TS(\mathcal{A}) = (Q, V)$ such that $Q = 2^S$ and $V = \{(X, res_{\mathcal{A}}(X) \cup C) \mid X, C \in 2^S\}$.*
2. *The* context-independent transition system *of $\mathcal{A}$ is an ordered pair $CITS(\mathcal{A}) = (Q, V)$ such that $Q = 2^S$ and $V = \{(X, res_{\mathcal{A}}(X)) \mid X \in 2^S\}$.*

The two transition systems from Definition 13 share their nodes, and $CITS(\mathcal{A})$ is a subgraph of $TS(\mathcal{A})$, but they are never equal (e.g., (S, S) is always a transition in $TS(\mathcal{A})$, but not in $CITS(\mathcal{A})$). Neither of the two transition systems is initialised, as any subset of the background set S can be the first state of an interactive process or a context-independent process of $\mathcal{A}$.

Following the well-established approach that the directed paths of a transition system provide the semantics of the dynamic system it is associated with, we can establish the soundness of the definitions of $TS(\mathcal{A})$ and $CITS(\mathcal{A})$ by relating the state sequences of the interactive processes of $\mathcal{A}$ to the state sequences of the two kinds of transition systems.

Proposition 7. *Let $\mathcal{A} = (S, A)$ be a reaction system and $X \subseteq S$.*

1. $sts(TS(\mathcal{A})) = STS(PROC(\mathcal{A}))$.
2. $sts(CITS(\mathcal{A})) = STS(CIPROC(\mathcal{A}))$.
3. $sts_X(TS(\mathcal{A})) = STS(PROC_X(\mathcal{A}))$.
4. $sts_X(CITS(\mathcal{A})) = STS(CIPROC_X(\mathcal{A}))$.

Proof. All statements follow directly from the definitions. □

Developing a satisfactory notion of a transition system capturing the behaviour of a reaction system is not a straightforward task. The reason is that in the standard treatment, the states of a transition system do not have any internal structure and the interactions between a system and its environment are represented as actions, such as messages or signals. In the case of reaction systems, however, the states do have structure (they are sets of entities) and the interaction with the environment (context) takes the form of sets of entities which are added to the current state of the system.

It is instructive to note that the state sequences defined by the directed paths of the (unlabelled) transition system $TS(\mathcal{A})$ record vertex based attributes rather than arc based attributes (as is usually the case in process algebras [26, 35]). This is natural for models of biological systems where the behaviours are perceived through changes of states and in general it is not possible to record the contexts thrown in by the environment.

8 Structuring context

In the literature on reaction systems, one mostly considers two extreme variants of context sequences, viz., there are either no restrictions imposed (as captured by the full transition systems of reaction systems), or no context is provided (as captured by the context-independent transition systems of reaction systems). A fundamental research topic is how to find interesting cases in-between these two extremes – in other words, the question is how to structure contexts? An example of an attempt to answer this question is provided by research on reaction systems with durations [6] where, for a given reaction system, the relevant context sequences are determined by a 'bigger' reaction system in which the original reaction system is embedded.

In this paper, we structure the environment and contexts it provides by assuming that it exhibits a finite-state behaviour. This will be captured by *context controllers* of two kinds: (1) controllers which throw in contexts irrespective of the current state of the reaction system (as proposed in [25] for model checking, where they were called context automata), considered in Section 8.1; and (2) more general controllers which throw in contexts depending also on the current system state of the reaction system (introduced for the first time in this paper), considered in Section 8.2.

8.1 State-oblivious context controllers

Definition 14. *A* state-oblivious context controller *over a background set S is an edge-labelled directed graph $\mathcal{E} = (Q, V)$ such that Q is a finite nonempty set and $V \subseteq Q \times 2^S \times Q$. We assume that, for every node $q \in Q$, there are $C \subseteq S$ and $q' \in Q$ such that $(q, C, q') \in V$.*

Note that a state-oblivious context controller over S is a labelled transition system over 2^S. The additional condition that each node of $\mathcal{E}$ has an outgoing edge means that $\mathcal{E}$ is always able to throw in a context.

Interactive processes generated under the control of state-oblivious context controllers are defined as follows.

Definition 15. *Let $\mathcal{A} = (S, A)$ be a reaction system, and $\mathcal{E} = (Q, V)$ a state-oblivious context controller over S.*
An interactive process in $\mathcal{A}$ controlled *by $\mathcal{E}$ and* initiated *at $(q, X) \in Q \times 2^S$ is an ordered pair $\pi = (\gamma, \delta)$ of finite sequences such that, for some $n \geq 1$, $\gamma = C_0, \ldots, C_n$ and $\delta = D_0, \ldots, D_n$, where $C_0, \ldots, C_n, D_0, \ldots, D_n \subseteq S$. Moreover, there are $q_0, \ldots, q_n \in Q$ satisfying:*

- *$(q, C_0, q_0) \in V$ and $D_0 = res_{\mathcal{A}}(X)$,*
- *$(q_i, C_{i+1}, q_{i+1}) \in V$ for $i = 0, \ldots, n-1$, and*
- *$D_i = res_{\mathcal{A}}(C_{i-1} \cup D_{i-1})$ for $i = 1, \ldots, n$.*

The state sequence of π is $STS(\pi) = X, W_0, W_1, \ldots, W_n$, where $W_i = C_i \cup D_i$ for $i = 0, \ldots, n$.

Note that the state sequence of π begins with X rather than W_0 as was the case previously. We refer to (q, X) as the *origin of* π. The intuition behind it is that when $\mathcal{A}$ begins in state X, the state of $\mathcal{E}$ is q.

We use $PROC_{(q,X)}(\mathcal{A}, \mathcal{E})$ to denote the set of all interactive processes in $\mathcal{A}$ controlled by $\mathcal{E}$ and initiated at (q, X), and $PROC(\mathcal{A}, \mathcal{E})$ to denote the set of all interactive processes in $\mathcal{A}$ controlled by $\mathcal{E}$, i.e.,

$$PROC(\mathcal{A}, \mathcal{E}) = \bigcup_{(q,X) \in Q \times 2^S} PROC_{(q,X)}(\mathcal{A}, \mathcal{E}) \ .$$

One can visualise the last definition using the progression in terms of columns now also incorporating the state of the context controller, following the origin (q, X) which provides the initial parameters in the last definition:

$$\begin{array}{cccccccc} & q & q_0 & q_1 & \ldots & q_{n-1} & q_n \\ \gamma: & & C_0 & C_1 & \ldots & C_{n-1} & C_n \\ \delta: & & D_0 & D_1 & \ldots & D_{n-1} & D_n \\ \tau: & X & W_0 & W_1 & \ldots & W_{n-1} & W_n \end{array} \tag{2}$$

Note that, for every $1 \leq k < n$:

$$((C_k, \ldots, C_n), (D_k, \ldots, D_n)) \in PROC_{(q_{k-1}, W_{k-1})}(\mathcal{A}, \mathcal{E}) \ .$$

In other words, a suffix of an interactive process is also an interactive process.

The motivation behind the above *extended* notion of interactive process (and one which is particularly needed in the case of state-aware context controllers introduced below) is to be able to have a clear concept of *extended* states and state sequences in which not only sets of entities are present, but also states of the controller. By an extended state sequence we mean $(q, X), W'_0, W'_1, \ldots, W'_n$, where $W'_i = (q_{i+1}, C_i \cup D_i)$ for $i = 0, \ldots, n$. Such a treatment also allows one to easily define a transition system representation of state sequences of interactive processes.

We introduce two kinds of transition systems for reaction systems controlled by state-oblivious context controllers. The first does not distinguish between entities thrown in by the environment and those produced by reaction system, and the second (to be used in Section 10) takes account of the contexts produced by controllers.

Definition 16. *Let $\mathcal{A} = (S, A)$ be a reaction system and $\mathcal{E} = (Q, V)$ be a state-oblivious context controller over S.*

1. *The* transition system *of $\mathcal{A}$ controlled by $\mathcal{E}$ is an ordered pair $TS(\mathcal{A}, \mathcal{E}) = (Q', U)$ such that*
   - *$Q' = Q \times 2^S$, and*
   - *U is the set of all ordered pairs $((q, X), (q', X')) \in Q' \times Q'$ such that there exists $(q, C, q') \in V$ satisfying $X' = C \cup res_{\mathcal{A}}(X)$.*
2. *The* labelled transition system *of $\mathcal{A}$ controlled by $\mathcal{E}$ is an ordered pair $LTS(\mathcal{A}, \mathcal{E}) = (Q', U)$ such that*

- $Q' = Q \times 2^S$, *and*
- U *is the set of all triples* $((q, X), C, (q', X')) \in Q' \times Q'$ *such that there exists* $(q, C, q') \in V$ *satisfying* $X' = C \cup res_{\mathcal{A}}(X)$.

For $TS(\mathcal{A}, \mathcal{E})$, a state sequence starting at a node (q, X) is obtained by taking the node sequence defined by a path originating at (q, X) and then deleting the first components (i.e., the states of $\mathcal{E}$). The set of all such sequences is denoted by $sts_{(q,X)}(TS(\mathcal{A}, \mathcal{E}))$, and then

$$sts(TS(\mathcal{A}, \mathcal{E})) = \bigcup_{(q,X) \in Q \times 2^S} sts_{(q,X)}(TS(\mathcal{A}, \mathcal{E}))$$

is the set of all state sequences of reaction system $\mathcal{A}$ controlled by $\mathcal{E}$.

Proposition 8. *Let* $\mathcal{A} = (S, A)$ *be a reaction system, and* $\mathcal{E} = (Q, V)$ *be a state-oblivious context controller over* S.

1. $sts_{(q,X)}(TS(\mathcal{A}, \mathcal{E})) = STS(PROC_{(q,X)}(\mathcal{A}, \mathcal{E}))$ *for every* $(q, X) \in Q \times 2^S$.
2. $sts(TS(\mathcal{A}, \mathcal{E})) = STS(PROC(\mathcal{A}, \mathcal{E}))$.

Proof. Both statements follow directly from the definitions. □

Both the full transition system of $\mathcal{A} = (S, A)$ and the context-independent transition system of $\mathcal{A}$ may, in essence, be obtained using suitable state-oblivious context controllers, as follows.

- Let $\mathcal{E}^{full} = (\{q\}, \{(q, C, q) \mid C \in 2^S\})$ be a state-oblivious context controller with exactly one state. Then $TS(\mathcal{A})$ can be obtained from $TS(\mathcal{A}, \mathcal{E}^{full})$ by replacing each node (q, X) by X.
- Let $\mathcal{E}^{cind} = (\{q\}, \{(q, \varnothing, q)\})$ be a state-oblivious context controller with exactly one state. Then $CITS(\mathcal{A})$ can be obtained from $TS(\mathcal{A}, \mathcal{E}^{cind})$ by replacing each node (q, X) by X.

It can be shown that the state sequences of transition systems defined as above coincide with those defined in the standard way.

8.2 State-aware context controllers

The totally arbitrary nature of the context sequences in interactive processes and, on the other hand, the extreme restriction of such sequences in context-independent interactive processes means that it is rather natural to allow and then analyse ways of varying the degree of such constraints. We have already made a first step in the previous sub-section. Now we will go one step further, by allowing a controller to access the current system state in order to decide what context to throw in. Such a feature is useful in modelling mutual interactions between (biological) systems and their surrounding environment.

Whereas a state-oblivious context controller had its 'own' set of states which were used to provide the desired contexts depending on its current state, a state-aware context controller will have states structured as ordered pairs (h, X), where

$h \in H$ is as previously a state of the controller, and X represents the current state of the reaction system being controlled. This simple device allows one to make the generation of contexts dependent on the current state of the reaction system.

Definition 17. *A* state-aware context controller *over a background set S is an edge-labelled directed graph $\mathcal{E} = (Q, V)$ such that, for some finite nonempty set H, $Q = H \times 2^S$ and $V \subseteq Q \times 2^S \times Q$, the following hold:*

- *For every $q \in Q$, there exist $C \subseteq S$ and $q' \in Q$ such that $(q, C, q') \in V$.*
- *If $((h, X), C, (h', X')) \in V$ then*
  - *$C \subseteq X'$, and*
  - *$((h, X), C, (h', C \cup X'')) \in V$, for every $X'' \subseteq S$.*

Intuitively, the first condition means that the controller always provides some context and the last condition means that the controller provides only contexts and cannot block reactions of any reaction system operating under it from producing their products.

The notion of an interactive process is now defined as follows.

Definition 18. *Let $\mathcal{A} = (S, A)$ be a reaction system, and $\mathcal{E} = (Q, V)$ be a state-aware context controller over S.*

An interactive process in $\mathcal{A}$ controlled *by $\mathcal{E}$ and* initiated *at $q = (h, X) \in Q$ is an ordered pair $\pi = (\gamma, \delta)$ of finite sequences such that, for some $n \geq 1$, $\gamma = C_0, \ldots, C_n$ and $\delta = D_0, \ldots, D_n$, where $C_0, \ldots, C_n, D_0, \ldots, D_n \subseteq S$. Moreover, there are $q_i = (h_i, C_i \cup D_i) \in Q$ for $i = 0, \ldots, n$ satisfying:*

- *$(q, C_0, q_0) \in V$ and $D_0 = res_{\mathcal{A}}(X)$.*
- *$(q_i, C_{i+1}, q_{i+1}) \in V$ for $i = 0, \ldots, n-1$.*
- *$D_i = res_{\mathcal{A}}(C_{i-1} \cup D_{i-1})$ for $i = 1, \ldots, n$.*

The state sequence of π is $STS(\pi) = X, W_0, W_1, \ldots, W_n$, where $W_i = C_i \cup D_i$ for $i = 0, \ldots, n$.

Note that (as was the case in Definition 15) the state sequence of π begins with X.

We use $PROC_q(\mathcal{A}, \mathcal{E})$ to denote the set of all interactive processes in $\mathcal{A}$ controlled by $\mathcal{E}$ and initiated at q, and $PROC(\mathcal{A}, \mathcal{E})$ to denote the set of all interactive processes in $\mathcal{A}$ controlled by $\mathcal{E}$, i.e.,

$$PROC(\mathcal{A}, \mathcal{E}) = \bigcup_{q \in Q} PROC_q(\mathcal{A}, \mathcal{E}) \ .$$

Transition systems of $\mathcal{A}$ controlled by $\mathcal{E}$ are defined analogously to those introduced for state-oblivious context controllers.

Definition 19. *Let $\mathcal{A} = (S, A)$ be a reaction system, and $\mathcal{E} = (Q, V)$ be a state-aware context controller over S.*

1. *The* transition system *of $\mathcal{A}$ controlled by $\mathcal{E}$ is an ordered pair $TS(\mathcal{A},\mathcal{E}) = (Q,U)$, where U is the set of all ordered pairs $((h,X),(h',X')) \in V$ such that there exists $((h,X),C,(h',X')) \in V$ satisfying $X' = C \cup res_{\mathcal{A}}(X)$.*
2. *The* labelled transition system *of $\mathcal{A}$ controlled by $\mathcal{E}$ is an ordered pair $LTS(\mathcal{A},\mathcal{E}) = (Q,U)$, where U is the set of all triples $((h,X),C,(h',X')) \in V$ such that $X' = C \cup res_{\mathcal{A}}(X)$.*

For $TS(\mathcal{A},\mathcal{E})$ and $q = (h,X) \in Q$, the sets $sts_q(TS(\mathcal{A},\mathcal{E}))$ and $sts(TS(\mathcal{A},\mathcal{E}))$ of state sequences are defined similarly as in the case of a transition system of $\mathcal{A}$ controlled by a state-oblivious context controller.

Proposition 9. *Let $\mathcal{A} = (S,A)$ be a reaction system, and $\mathcal{E} = (Q,V)$ be a state-aware context controller over S.*

1. *$sts_q(TS(\mathcal{A},\mathcal{E})) = STS(PROC_q(\mathcal{A},\mathcal{E}))$ for every $q \in Q$.*
2. *$sts(TS(\mathcal{A},\mathcal{E})) = STS(PROC(\mathcal{A},\mathcal{E}))$.*

Proof. Both statements follow directly from the definitions. □

State-aware context controllers are more general than the state-oblivious ones, in the sense that the former can be used to simulate the latter. Consider, for example, a reaction system $\mathcal{A} = (S,A)$ controlled by a state-oblivious context controller $\mathcal{E} = (Q,V)$. Then $sts(TS(\mathcal{A},\mathcal{E})) = sts(TS(\mathcal{A},\mathcal{E}'))$, where $\mathcal{E}' = (Q \times 2^S, V')$ is a state-aware context controller with V' comprising all triples $((q,X),C,(q',X' \cup C))$ such that $X,X' \subseteq S$ and $(q,C,q') \in V$.

9 Equivalences based on transition systems

The full transition systems associated with reaction systems lead in a natural way to an equivalence notion:

Definition 20. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set. Then $\mathcal{A}$ and $\mathcal{B}$ are* transition system equivalent*, denoted by $\mathcal{A} \sim_{TS} \mathcal{B}$, if $TS(\mathcal{A}) = TS(\mathcal{B})$.*

Both $\sim_{PROC}$ and $\sim_{TS}$ are equivalences defined in terms of states. They are based on manifestations of behaviour expressed in terms of interactive processes and transition systems respectively. In general for a given reaction system, there will be loss of behavioural information when one focusses solely on the state sequences of its interactive processes, or of its transition system: in general it will not be clear for a given state, what has been produced from a previous state and what has been thrown in as context. We will later remove this restriction and consider labelled transition systems associated with reaction systems.

Clearly, $\sim_{TS}$ is an equivalence relation. Moreover, it turns out that two reaction systems that have the same associated transition system, are also process equivalent and, moreover, are equivalent in the classical sense.

Proposition 10. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set. Then $\mathcal{A} \sim \mathcal{B}$ if and only if $\mathcal{A} \sim_{PROC} \mathcal{B}$ if and only if $\mathcal{A} \sim_{TS} \mathcal{B}$.*

Proof. Follows immediately from Corollary 1, Proposition 7, and the definition of the set of state sequences $sts(\mathcal{T})$ of a transition system $\mathcal{T}$. □

10 Bisimulation-based behavioural equivalence

In the previous sections, transition systems associated with reaction systems as well as the environments in which they operate were basically concerned with state sequences. This means, implicitly, that we made no distinction between entities produced by reaction systems and those in contexts thrown-in by the environment. We will now reassess the way of observing behaviours and introduce new equivalences between reaction systems.

We base our discussion in this section on the well-known notion of bisimulation between transition systems. Such a notion has been successfully applied to capture a range of equivalence notions for communicating computing systems ([26, 27, 35]). A standard way of introducing bisimulation-based equivalence is to consider two labelled transition systems, $LTS = (Q, V)$ and $LTS' = (Q', V')$, and then find a *bisimulation* relation $B \subseteq Q \times Q'$ aiming at identifying equivalent (or *bisimilar*) states. What is required is that if $(q, q') \in B$, then all transitions from q can be simulated by transitions from q' which lead to bisimilar states, and vice versa (i.e., if $(q, a, r) \in V$, then there is a $(q', a, r') \in V'$ such that $(r, r') \in B$, and vice versa). The notion of equivalence obtained in this way does not depend on a specific relation B as there is always the largest bisimulation[5] relation for a given pair of transition systems. Moreover, two initialised transition systems are bisimulation equivalent if their initial states are bisimilar.

We now make two general observations. First, bisimulation-based equivalences are usually label-based, and the states of the transition systems have no structure. This, of course, is no longer the case for reaction system behaviours, where states are sets of entities. Second, since we decided to accept that contexts thrown-in by the environment can be observed, the arcs of transition systems representing reaction systems and their environments should now be labelled with sets of entities. As a consequence of these two observations, in this section we will use labelled transition systems rather than transition systems of reaction system controlled by context controllers.

By using bisimulation-based equivalence, we expect to capture precisely what it means that two systems 'react' in an 'equivalent' manner to stimuli provided by a given context controller.

The next definition considers two reaction systems and a context controller over a common background set. Moreover, a set Y identifies entities considered as 'observable'.

[5] Since the union of two bisimulation relations is also a bisimulation relation, there is always the largest (in the set inclusion sense) bisimulation relation for a given pair of transition systems.

Definition 21. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S, $\mathcal{E}$ be a state-oblivious or state-aware context controller over S, and $Y \subseteq S$.*

A Y-bisimulation *for $LTS(\mathcal{A},\mathcal{E}) = (Q,V)$ and $LTS(\mathcal{B},\mathcal{E}) = (Q,U)$ is a relation $B \subseteq Q \times Q$ such that if $((h,X),(h',X')) \in B$, then the following hold:*

- $X \cap Y = X' \cap Y$.
- *If $((h,X),C,(\widehat{h},\widehat{X})) \in V$, then there is a $((h',X'),C',(\widehat{h}',\widehat{X}')) \in U$ such that $C \cap Y = C' \cap Y$ and $((\widehat{h},\widehat{X}),(\widehat{h}',\widehat{X}')) \in B$.*
- *If $((h',X'),C',(\widehat{h}',\widehat{X}')) \in U$, then there is a $((h,X),C,(\widehat{h},\widehat{X})) \in V$ such that $C \cap Y = C' \cap Y$ and $((\widehat{h},\widehat{X}),(\widehat{h}',\widehat{X}')) \in B$.*

The above formalisation of bisimulation-based equivalence leads to a range of meaningful equivalences based on the idea of bisimulation, which are capable of capturing subtle aspects of the 'reactiveness' of $\mathcal{A}$ and $\mathcal{B}$.

Proposition 11. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S, $\mathcal{E}$ be a state-oblivious or state-aware context controller over S, and $Y \subseteq S$. Then there exists a Y-bisimulation $B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}}$ for $LTS(\mathcal{A},\mathcal{E})$ and $LTS(\mathcal{B},\mathcal{E})$ such that $B \subseteq B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}}$, for every Y-bisimulation B for $LTS(\mathcal{A},\mathcal{E})$ and $LTS(\mathcal{B},\mathcal{E})$*

Proof. Follows directly from the definitions. □

Note that $B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}}$ is well-defined since $B = \varnothing$ is always a Y-bisimulation.

Thus $B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}}$ is the largest Y-bisimulation for $LTS(\mathcal{A},\mathcal{E})$ and $LTS(\mathcal{B},\mathcal{E})$. It allows one to establish, for example, which initial states for $\mathcal{A}$ and $\mathcal{B}$ controlled by $\mathcal{E}$ are equivalent (bisimilar) when one is interested in obtaining equivalent reactive behaviour assuming that only entities in Y are observable.

Proposition 12. *Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be reaction systems with the same background set S, $\mathcal{E}$ be a state-oblivious or state-aware context controller over S, and $Y \subseteq S$. Then $(q,q'') \in B^Y_{\mathcal{A},\mathcal{C},\mathcal{E}}$, for all $(q,q') \in B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}}$ and $(q',q'') \in B^Y_{\mathcal{B},\mathcal{C},\mathcal{E}}$.*

Proof. Follows directly from the definitions. □

The above result states that bisimilarity is transitive while the next one states that it is preserved by making the set of observable entities smaller.

Proposition 13. *Let $\mathcal{A}$ and $\mathcal{B}$ be reaction systems with the same background set S, $\mathcal{E}$ be a state-oblivious or state-aware context controller over S, and $Z \subseteq Y \subseteq S$. Then $B^Y_{\mathcal{A},\mathcal{B},\mathcal{E}} \subseteq B^Z_{\mathcal{A},\mathcal{B},\mathcal{E}}$.*

Proof. Follows directly from the definitions. □

Note that the notion of bisimulation-based equivalence introduced in this section is immediately extendable to equivalence based on a set of context controllers rather than a single one.

11 Discussion

In this paper we proposed a framework for considering equivalences of reaction systems. The following aspects of this paper are 'immediate' candidates for a follow-up line of research.

(1) We have established a new bridge between the theory of reaction systems and the theory of transition systems by proposing various representations of reaction systems by transition systems. This should lead to a transfer of many methods of investigating the behaviour of dynamic systems through transition systems into the domain of reaction systems. Among the relevant specific aspects of theory of transition systems that would be good candidates for a transfer into the realm of reaction systems are: various variants of weak bisimulation equivalence and testing equivalence, and efficient verification techniques and tools.

(2) We introduced a formalisation of finite-state environments through state-oblivious and state-aware context controllers. This naturally leads to a novel line of research concerning properties of state sequences. The investigation of properties of state sequences of context-independent processes of reaction systems resulted in a rich and elegant theory (see, e.g., [9, 15, 20, 30, 31, 33, 34]). Reaction systems with context determined by context controllers form a natural next level of a systematic and thorough investigation of state sequences (clearly, not much can be said when context is arbitrary as in the general model of reaction systems).

(3) In considering equivalences defined directly on reaction systems, functional equivalence played a central role in this paper. In [10] we considered more subtle *enabling equivalence*. Incorporating also this equivalence in our framework is certainly a very natural research topic.

(4) The study of model checking techniques for reaction systems controlled by state-oblivious context controllers has been initiated in [25]. In this paper, we introduced much more general framework of reaction systems controlled by state-aware context controllers, the ensuing model checking problem is both important and challenging.

Acknowledgements

The authors are indebted to Robert Brijder and the anonymous referee for useful comments on this paper.

References

1. Arnold, A. (1994) *Finite Transition Systems: Semantics of Communicating Systems*, Prentice Hall.
2. Azimi, S., Iancu, B., Petre, I. (2014) Reaction system models for the heat shock response. *Fundamenta Informaticae* **131**, 1–14.
3. Azimi, S., Panchal, C., Czeizler, E., Petre, I. (2015) Reaction systems models for the self-assembly of intermediate filaments. *Annals of University of Bucharest* **LXII(2)**, 9–24.

4. Baier, C., Katoen, J.-P. (2008) *Principles of model checking*. The MIT Press.
5. Brijder, R., Ehrenfeucht, A., Main, M., Rozenberg G. (2011) A tour of reaction systems. *International Journal of Foundations of Computer Science* **22(07)**, 1499–1517.
6. Brijder, R., Ehrenfeucht, A., Rozenberg, G. (2011) Reaction systems with duration. *Lecture Notes in Computer Science* **6610**, Springer-Verlag, 191–202.
7. Corolli, L., Maj, C., Marini, F., Besozzi, D., Mauri, G. (2012) An excursion in reaction systems: From computer science to biology. *Theoretical Computer Science* **454** 95–108.
8. Dennunzio, A., Formenti, E., Manzoni, L. (2014) Extremal combinatorics of reaction systems. In *International Conference on Language and Automata Theory and Applications*, Springer-Verlag, 297–307.
9. Dennunzio, A., Formenti, E., Manzoni, L., Porreca, A.E. (2015) Ancestors, descendants, and gardens of eden in reaction systems. *Theoretical Compututer Science* **608**, 16–26.
10. Ehrenfeucht, A., Kleijn, J., Koutny, M., Rozenberg G. (2017) Evolving reaction systems. *Theoretical Compututer Science* **682**, 79–99.
11. Ehrenfeucht, A., Kleijn, J., Koutny, M., Rozenberg G. (2011) Modelling reaction systems with Petri nets. In *BioPPN-2011, International Workshop on Biological Processes & Petri Nets*, CEUR-WS Workshop Proceedings, 36–52.
12. Ehrenfeucht, A., Kleijn, J., Koutny, M., Rozenberg G. (2012) Reaction Systems: A Natural Computing Approach to the Functioning of Living Cells. In: Hector, Z. (Ed.), *A Computable Universe: Understanding and Exploring Nature as Computation*, Chapter 10, World Scientific, Singapore, 189–208.
13. Ehrenfeucht, A., Main, M.G., Rozenberg, G. (2011) Functions defined by reaction systems. *International Journal of Foundations of Computer Science* **22**, 167–178.
14. Ehrenfeucht, A., Main, M.G., Rozenberg, G. (2010) Combinatorics of life and death for reaction systems. *International Journal of Foundations of Computer Science* **21**, 345–356.
15. Ehrenfeucht, A., Main, M., Rozenberg G., Thompson Brown, A. (2012) Stability and chaos in reaction systems. *International Journal of Foundations of Computer Science* **23**, 1173–1184.
16. Ehrenfeucht, A., Petre, I., Rozenberg, G. (2017) Reaction systems: A model of computation inspired by the functioning of the living cell. In: Konstantinidis, S., Moreira, N., Reis, R., Shallit, J. (Eds.) *The Role of Theory in Computer Science* World Scientific, 1–32.
17. Ehrenfeucht, A. Rozenberg, G. (2007) Reaction systems. *Fundamenta Informaticae* **75**, 263–280.
18. Ehrenfeucht, A. Rozenberg, G. (2007) Events and modules in reaction systems. *Theoretical Compututer Science* **376**, 3–16.
19. Ehrenfeucht, A. Rozenberg, G. (2009) Introducing time in reaction systems. *Theoretical Compututer Science* **410**, 310–322.
20. Genova, D., Hoogeboom, H.J., Jonoska, N. (2017) A graph isomorphism condition and equivalence of reaction systems. *Theoretical Compututer Science* **701**, 109–119.
21. Hopcroft, J., Motwani, R., Ullman, J. (2006) *Introduction to Automata Theory, Languages, and Computation*, Prentice Hall.
22. Kari, L., Rozenberg, G. (2008) The many facets of natural computing. *Communications of the ACM* **51**, 72–83.
23. Keller, R.M. (1976) Formal Verification of Parallel Programs. *Communications of the ACM* **19**, 371–384.

24. Lehninger, A. (1965) *Bioenergetics: The Molecular Basis of Biological Energy Transformations*, W.A. Benjamin, Inc., New York, Amsterdam.
25. Meski, A., Penczek, W., Rozenberg, G. (2015) Model checking temporal properties of reaction systems. *Information Sciences* **313**, 22–42.
26. Milner, R. (1989) *Communication and Concurrency*. PHI Series in Computer Science, Prentice Hall.
27. Park, D. (1981) Concurrency and Automata on Infinite Sequences. *Lecture Notes in Computer Science* **104**, Springer-Verlag, 167–183.
28. Reisig, W. (1985) *Petri Nets (an Introduction)*. EATCS Monographs on Theoretical Computer Science, Springer-Verlag, Heidelberg.
29. Rozenberg, G., Bäck, T.,Kok, J. (2012) *Handbook of Natural Computing*, Springer-Verlag.
30. Salomaa, A. (2012) Functions and sequences generated by reaction systems. *Theoretical Compututer Science* **466**, 87–96.
31. Salomaa, A. (2012) On state sequences defined by reaction systems. *Lecture Notes in Computer Science* **7230**, Springer-Verlag, 271–282.
32. Salomaa, A. (2013) Functional constructions between reaction systems and propositional logic. *International Journal of Foundations of Computer Science* **24**, 147–160.
33. Salomaa, A. (2013) Minimal and almost minimal reaction systems. *Natural Computing* **12**, 369–376.
34. Salomaa, A. (2015) Two-step simulations of reaction systems by minimal ones. *Acta Cybernetica* **22**, 393–311.
35. Sangiorgi, D. (2011) *Introduction to Bisimulation and Coinduction*, Cambridge University Press.